CS360 Midterm 2– Spring, 2016 – James S. Plank – March 10, 2016

In all of these questions, please assume the following:

- Pointers and longs are 4 bytes.
- The machine is little endian, but that doesn't matter in any of these questions.
- There are no segmentation violations or bus errors in any of this code.
- Any assembly code questions use the **jassem** assembly code.
- Any assembly code that corresponds to compiled C code is unoptimized.

Question 1:

The program on the answer sheet for Question 1 reads 16-character strings from standard input, and prints the first 10,000 of them that contain the letter 'A'. This program runs correctly, in that it gives the proper output. However, it has three serious problems with it. Your job is to:

1. Identify the three problems by circling the offending lines of code for each and labeling it A, B and C.

2. For each of A, B, and C, explain why it is a problem. Don't just say, for example, "runs too slowly." Instead, explain why it runs slowly.

3. Rewrite the program so that it does not have any of these problems.

Question 2:

When I type the command "ls */f1.txt" into the shell, it lists all files named "f1.txt" that are in sudirectories of the current directory. I want you to write lsf1.c, which works just like "ls */f1.txt". If there are no files named f1.txt in the subdirectories of the current directory, simply print "None Found." Here are examples.

```
UNIX> ls
Dir1  Dir2  Dir3
UNIX> ls Dir1
f1.txt      f2.txt      f3.txt
UNIX> ls Dir2
f0.txt      f1.txt
UNIX> ls Dir3
Sub   f0.txt      f2.txt
UNIX> ls Dir3/Sub
f1.txt
UNIX> lsf1
Dir1/f1.txt Dir2/f1.txt
UNIX> cd Dir3
UNIX> lsf1
Sub/f1.txt
UNIX> cd ../Dir2
UNIX> lsf1
None Found
UNIX>
```

If you are strapped for time, write down your approach to solving it. You'll get some partial credit for that.

Question 3:

When we compile and run the program below, the first four lines of output are:

```
p:    0x100130
p[0]: 0x100140
p[1]: 0x100150
p[2]: 0x100160
```

What are the last six lines of output?

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef unsigned long UL;

main()
{
  int **p, *q, *r;
  int i, j;

  p = (int **) malloc(sizeof(int *) * 3);
  for (i = 0; i < 3; i++) {
    p[i] = (int *) malloc(sizeof(int) * 3);
  }

  printf("p:    0x%lx\n", (UL) p);
  for (i = 0; i < 3; i++) {
    printf("p[%d]: 0x%lx\n", i, (UL) p[i]);
    for (j = 0; j < 3; j++) {
      p[i][j] = 256+i*16+j;
    }
  }

  q = *(p+1);
  q++;
  r = *p;

  printf("\n");

  printf("A: 0x%lx\n", (UL) (p+1));
  printf("B: 0x%lx\n", (UL) q);
  printf("C: 0x%lx\n", (UL) *q);
  printf("D: 0x%lx\n", (UL) *(q+1));
  printf("E: 0x%lx\n", (UL) r);
  printf("F: 0x%lx\n", (UL) r[9]);
}
```

Question 4

Suppose that when this procedure runs, the value of **fp** is 0x16780. Also, the **ret** statement in **proc** is at memory address 0x1234.

```
void proc(int a, int b)
{
  int c[3];

  d(b, 15);
  return 4;
}
```

Please answer the following questions.
For parts E through J, I want numbers, not variables or expressions.

Part A: What is the first line of assembly code in **proc**?
Part B: What is the second line?
Part C: What is the assembly instruction at address 0x122c?
Part D: What is the assembly instruction at address 0x1230?
Part E: What is the value of **sp** right after the **jsr** call returns?
Part F: What is the address of **a**?
Part G: What is the address of **b**?
Part H: What is the address of **c[1]**?
Part I: When **proc** returns, what is the value in memory address 0x16768?
Part J: At what address can we find the value that the **pc** will be after proc returns?

Question 5

Please convert the following C procedure into assembly code:

```
void a(int b)
{
  int i;

  while (c() + d() > 0) b += 5;
  return b;
}
```

Question 6

Please convert the last line (the return statement) of the following C procedure into assembly code:

```
void a(int **p)
{
  int i;

  sscanf("%d", &i);
  return *(p[i]);
}
```

Handy prototypes:

```
mempcy(char *to, char *from, int nbytes)
char *strchr(char *s, char tofind);

int open(char *path, int flags [ , int mode ] )
int close(int fd)
ssize_t read(int fd, void *buf, size_t count)
ssize_t write(int fd, const void *buf, size_t count)

FILE *fopen(char *path, char *mode);
int fclose(FILE *f);
int fread(void *ptr, int size, int nbytes, FILE *f)
int fwrite(void *ptr, int size, int nbytes, FILE *f)

int S_ISDIR(mode_t mode) /* Is the file a directory? */

struct stat {
   mode_t   st_mode;   /* File mode (see mknod(2)) */
   ino_t    st_ino;    /* Inode number */
   /* other stuff omitted */
};

struct dirent {
   char *de_name;
   /* other stuff omitted */
}
```

More Handy prototypes:

```
strcpy(char *to, char *from)
strcat(char *to, char *from)
char *strdup(char *s)


DIR *opendir(char *path)
int closedir(DIR *d);
struct dirent *readdir(DIR *D);

int stat(char *path, struct
    stat buf);  /* returns zero
                   on success */
```

CS360 Midterm – Answer Sheet

Name: ______________________

Email: ___________@vols.utk.edu

Put your output one character per underlined place.

Question 1

Here's the program – circle the offending lines and label them.

```
main()
{
  char s[160001];
  char buf[17];
  int n;

  s[0] = '\0';
  n = 0;

  while (n < 10000 && read(0, buf, 16) == 16) {
    buf[16] = '\0';
    if (strchr(strdup(buf), 'A') != NULL) {
      n++;
      strcat(s, buf);
    }
  }

  write(1, s, n*16);
  exit(0);
}
```

Explain why A, B and C are bad below:

Rewrite the program:

CS360 Midterm – Answer Sheet

Name: ________________________

Email: ____________@vols.utk.edu

Question 2

Write the code here. Don't bother with include statements.

CS360 Midterm – Answer Sheet

Name: ______________________

Email: ___________@vols.utk.edu

Question 3

Write the output here:

A:
B:
C:
D:
E:
F:

Question 4

Write your answers here

Part A:
Part B:
Part C:
Part D:
Part E:
Part F:
Part G:
Part H:
Part I:
Part J:

Question 5

Question 6